第2章

CASLモード

1．アセンブラ言語

コンピュータ言語においてＢＡＳＩＣ，ＦＯＲＴＲＡＮ，ＣＯＢＯＬといった高水準な言語は、その記述がわかりやすいためプログラミングが容易であり、開発期間も短くて済みます。しかしこれらは、メモリの作業領域を多く必要としたり処理速度が遅いといった面があります。コンピュータの性能を最も効率よく活用できるのは機械語ですが、これは機械語コードで入力する必要があり、プログラミングは非常に手間がかかります。アセンブラはこの機械語に最も近い言語で、ＢＡＳＩＣほど手軽ではないにしても、機械語に比べて使い易く、コンピュータの性能を限界近くまで活用することができます。ただ、アセンブラ言語で効率の良いプログラムを作るには、各命令を実行するときに、レジスタやフラグなどが、どのような働き、動きをするのかを知る必要があります。これらの働き、動きなどはコンピュータのハードウェアに大きく左右されメーカーや機種により異なるのが普通です。

このため通産省（通商産業省）の情報処理技術者試験では、ＣＯＭＥＴという仮想の計算機を設定し、これのアセンブラ言語としてＣＡＳＬの仕様を取り決めた上で試験問題を出します。

本計算機のＣＡＳＬモードはこのＣＡＳＬの仕様に準拠したアセンブラ言語が扱えるため、情報処理技術者試験（第１種では必須、第２種では選択）のトレーニングとしてプログラミングからデバッグそしてシミュレーションまでの学習が容易におこなえます。また、小型軽量であるため、ご家庭でも、学校、職場、あるいは電車や車の中ででも手軽にお使いいただけます。

また、ＣＡＳＬは従来の情報処理技術者試験で出題されていたアセンブラ言語、ＣＡＰ－Ｘと比較しますと、アドレスの与え方が豊富になり、またスタック、入出力といった機能が付加され、最近の計算機の実情に合った仕様になっており、アセンブラ言語の入門用としても適しています。

本書は、本計算機の操作方法に重点をおいて説明していますので、ＣＯＭＥＴやＣＡＳＬの仕様、ＣＡＳＬの文法、プログラミングについてのくわしい説明は、有名書店で数多く販売されている、情報処理技術者試験のＣＡＳＬに関する書籍をご参照ください。

1.1　COMETの仕様概略

アセンブラ言語の理解を深めるため、本機のCASLモードでの仮想ハードウェア仕様（COMETに準拠）について概略を記しておきます。

(1) **1語長**　：16ビット

(2) **制御方式**　：逐次制御方式（ノイマン型）

(3) **数値の表現**：16ビットの2進数、負数は2の補数表示

(4) **レジスタ**　：

① GR0～4（16ビット）、Gneral Register 汎用レジスタ、GR1～GR4は、指標レジスタとしても使用。さらにGR4はスタックポインタとしても使用される。

② PC（16ビット）、Program Counter 実行中の命令語の先頭アドレスを保持するレジスタ。

③ FR（2ビット）、Flag Register 演算、比較結果の情報保持レジスタ。

	GRに設定されたデータ		
	負	ゼロ	正
FRの値	10	01	00

1.2　命令語の構成

CASLの命令は2語長と定義されています。その構成をまとめます。

第1語				第2語	命令語とアセンブラの対応	
OP		GR	XR	adr		ニモニックスペル
主	副					
0	0				未　使　用	
1	0 1 2				LD　GR,adr,XR ST　GR,adr,XR LEA　GR,adr,XR	load store load effective address
2	0 1				ADD GR,adr,XR SUB GR,adr,XR	add arithmetic subtract arithmetic
3	0 1 2				AND GR,adr,XR OR　GR,adr,XR EOR GR,adr,XR	and or exclusive or

第1語				第2語	命令語とアセンブラの対応	
OP						~
主	副	GR	XR	adr		ニモニックスペル
4	0				CPA GR,adr,XR	compare arithmetic
	1				CPL GR,adr,XR	compare logical
5	0				SLA GR,adr,XR	shift left arithmetic
	1				SRA GR,adr,XR	shift right arithmetic
	2				SLL GR,adr,XR	shift left logical
	3				SRL GR,adr,XR	shift right logical
6	0	――			JPZ adr,XR	jump on plus or zero
	1	――			JMI adr,XR	jump on minus
	2	――			JNZ adr,XR	jump on non zero
	3	――			JZE adr,XR	jump on zero
	4	――			JMP adr,XR	unconditional jump
7	0	――			PUSH adr,XR	push effective address
	1			―――	POP GR	pop up
8	0	――			CALL adr,XR	call subroutin
	1	――		―――	RET	returu from subroutin
9 ∫ F					未 使 用	

1.3 命令の種類と機能

本機はCASLで定義された23の命令をもっています。以下に各命令の機能を説明します。なお、説明には次の表記法を使います。

① GR ：GRの値を番号とする汎用レジスタを示します。
（ただし0≦GR≦4の範囲）

② XR ：XRの値を番号とする指標レジスタを示します。
（ただし1≦XR≦4の範囲）

③ SP ：スタックポインタを示します。

④ adr ：ラベル名に対応する番地、または10進の定数（ただし−32768≦adr≦65535）を示します。adrはアドレスとして0〜65535の値を持ちますが、32768〜65535の値を負の10進定数で記述することもできます。（例：65535番地は−1と記述できます）

⑤　実効アドレス：ａｄｒとＸＲの内容とのアドレス加算値、またはその値が示す番地です。
⑥　（Ｘ）　：Ｘ番地の内容、またはＸが汎用レジスタ（ＧＲ）を示す場合はそのＧＲの内容を表します。
⑦　［　］　：［　］に囲まれた部分は省略可能です。ＸＲを省略した場合は、指標レジスタによる修飾はおこなわれません。

[各命令と機能]

(1) **LD　GR, adr [, XR]**

機能：（実効アドレス）をＧＲに設定します。

(2) **ST　GR, adr [, XR]**

機能：（ＧＲ）を実効アドレスが示す番地に格納します。

(3) **LEA　GR, adr [, XR]**

機能：実効アドレスをＧＲに設定します。このときのＧＲの値によりＦＲが設定されます。

例　：ＬＥＡ　ＧＲ１，１∅　　ＧＲ１に１∅を設定します。
ＬＥＡ　ＧＲ２，∅，ＧＲ３　　ＧＲ３をＧＲ２に移動します。
ＬＥＡ　ＧＲ３，－１，ＧＲ１　　ＧＲ１から１を減じた値をＧＲ３に設定します。

(4) **ADD　GR, adr [, XR]**

機能：算術加算命令で、（ＧＲ）と（実効アドレス）の加算結果をＧＲに設定します。このときの演算結果によりＦＲが設定されます。

例　：ＡＤＤ　ＧＲ２，ＷＲＫ　　（ＷＲＫ番地）を（ＧＲ２）に加算します。
ＡＤＤ　ＧＲ１，Ａ１，ＧＲ３　　Ａ１とＧＲ３とのアドレス加算結果の内容をＧＲ１に加算します。

(5) **SUB　GR, adr [, XR]**

機能：ＡＤＤ命令の逆で、（ＧＲ）と（実効アドレス）の減算結果をＧＲに設定します。演算結果によりＦＲが設定されます。

(6) AND GR, adr [, XR]

(7) OR GR, adr [, XR]

(8) EOR GR, adr [, XR]

機能：1語16ビットのビットごとの論理積（AND）、論理和（OR）、排他的論理和（EOR）を（GR）と（実効アドレス）で行い、GRに設定します。演算結果によりFRが設定されます。

例 ：AND GR2, MASK7　　MASK7を7にしておくと、GR2の下位3ビットが取り出せます。

EOR GR1, FF　　FFが#FFFF（－1）の場合、GR1の全ビットが反転します。

(9) CPA GR, adr [, XR]

(10) CPL GR, adr [, XR]

機能：（GR）と（実効アドレス）の算術比較（CPA）または論理比較（CPL）を行い、比較結果によりFRにつぎの値を設定します。

比較結果	FRの設定ビット値
（GR）＞（実効アドレス）	00
（GR）＝（実効アドレス）	01
（GR）＜（実効アドレス）	10

(11) SLA GR, adr [, XR]

機能：（GR）を符号を除き実効アドレスで指定したビット数だけ左シフトを行います。シフトの結果、空いたビット位置には0が入り、FRを設定します。

(12) SRA GR, adr [, XR]

機能：（GR）を符号を除き実効アドレスで指定したビット数だけ右シフトを行います。シフトの結果、空いたビット位置には符号と同じものが入り、FRを設定します。

(13) SLL GR, adr [, XR]

(14) SRL GR, adr [, XR]

機能：（GR）を符号を含み実効アドレスで指定したビット数だけ左シフト（SLL）、または右シフト（SRL）を行います。シフトの結果、空いたビットには0が入り、FRを設定します。

15) JPZ　adr [, XR]

16) JMI　adr [, XR]

17) JNZ　adr [, XR]

18) JZE　adr [, XR]

機能：FRの値により実効アドレスに分岐します。分岐しないときは次の命令に進みます。

命　令	分岐するときのFR値	意　味
JPZ	00, 01	正または0のとき分岐
JMI	10	負のとき分岐
JNZ	00, 10	0でないとき分岐
JZE	01	0のとき分岐

19) JMP　adr [, XR]

機能：FRの値に関係なく無条件に実効アドレスに分岐します。

20) PUSH　adr [, XR]

機能：SPから1をアドレス減算してから実効アドレスを（SP）番地に格納します。

21) POP　GR

機能：（SP）番地の内容をGRに設定してからSPに1をアドレス加算します。

22) CALL　adr [, XR]

機能：サブルーチンへジャンプする命令です。サブルーチンでの実行完了後の戻り番地確保のためにスタックを用います。動作としてはSPから1をアドレス減算してからPCの現在値に2をアドレス加算した値を（SP）番地に格納し、実効アドレスに分岐します。

23) RET

機能：CALLに対する戻り命令です。（SP）番地の内容を取り出してからSPに1をアドレス加算し、先に取り出した番地に分岐します。

1.4 アセンブラの文法

本機のアセンブラ言語はCASLと同様に、疑似命令、マクロ命令、機械語命令を持っています。
疑似命令、マクロ命令、機械語命令は次のように記述します。

ラベル欄	命令コード欄	オペランド欄	注釈欄
[ラベル]	START	[実行開始番地]	[注　釈]
空　白	END	空　　白	[注　釈]
[ラベル]	DC	定　　数	[注　釈]
[ラベル]	DS	領域の語数	[注　釈]
[ラベル]	IN	入力領域、入力文字長	[注　釈]
[ラベル]	OUT	出力領域、入力文字長	[注　釈]
[ラベル]	EXIT	空　　白	[注　釈]
[ラベル]	機械語命令	[命令の種類と機能] 参照	[注　釈]

表中、「空白」は記入してはならない箇所で、[　] で囲まれた部分は省略可能を意味します。

1.5 疑似命令

疑似命令はアセンブラの制御、定数の定義、プログラムの連結のために必用なデータの生成などを行うもので、次の4種があります。

(1) **START**：プログラムの先頭、実行開始番地の定義や他のプログラムとの連結のための入口名の定義をします。プログラムの最初には必要です。

(2) **END**　：プログラムの終りの定義をします。プログラムの最後に必要です。

(3) **DC**　：define constant の略で、定数で指定したデータを格納します。定数には次の4種があります。

① 10進定数　　DC n

nで指定した10進数値を1語の2進データとして格納します。
nは−32768～32768の範囲で、この範囲を越える指定のときは下位16ビットが格納されます。

② １６進定数　　DC　#h

hは4桁の１６進数（0～F）です。hで指定された１６進数値を１語の２進数値データとして格納します。hは#ØØØØ≦h≦#FFFFの範囲です。なお本機では#の代りに&でも同結果が得られます。

③ 文字定数　　DC　'文字列'

文字列の左端から１文字ずつ連続する語の下位8ビット（第8～１５ビット）に文字データを格納します。各語の第0～7ビットには0のビットが入ります。文字列には276頁のキャラクタコード表のうちで、アポストロフィ（'）、π、√￣を除く文字が使えます。なお文字列の長さは0であってはなりません。

④ アドレス定数　DC　ラベル名

ラベル名に対応するアドレス値を１語の２進数として格納します。

(4) **DS**　：define storageの略で指定した語数の領域を確保します。

領域の語数は、１０進定数（≧0）で指定します。0の場合、領域は確保されず、ラベル名のみが有効となります。

1.6　マクロ命令

(1) **IN**　：あらかじめ割り当てた入力装置から入力領域に、１レコードの文字データを入力します。本機ではカード読み取り装置からのデータを模擬的にキーから入力できます。なお、EOF（end of file）の入力は [EOF ↓] キーを用います。

(2) **OUT**　：出力領域に格納されている文字データを１レコードとして、あらかじめ割り当てた出力装置に出力します。本機では表示あるいはプリンタに出力できます。

(3) **EXIT**　：プログラムの実行を終了します。

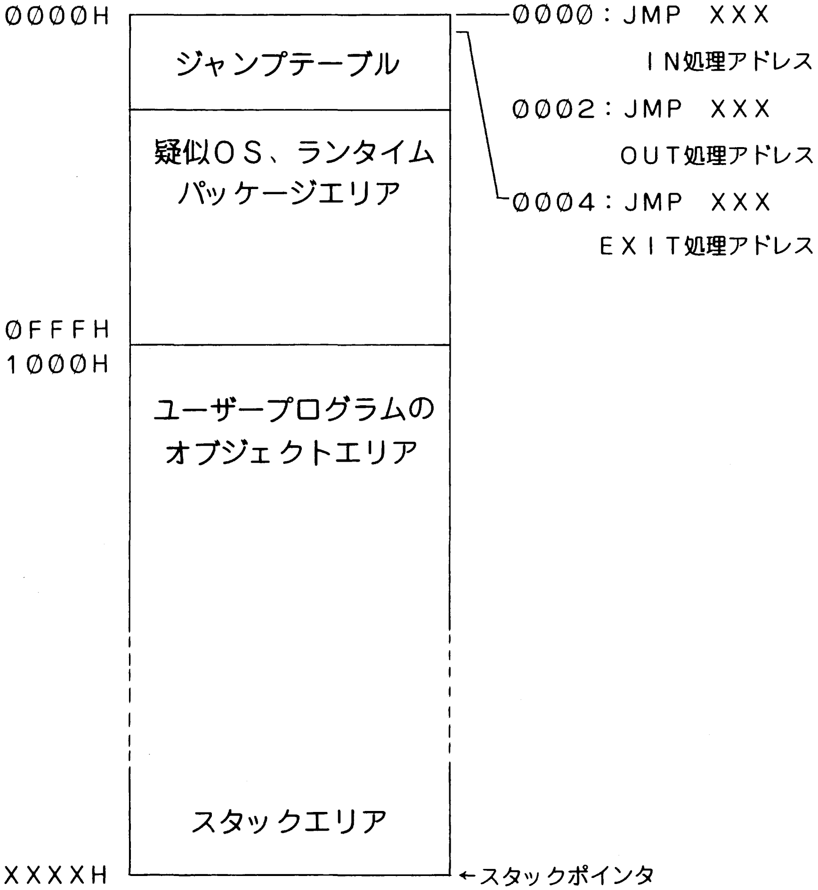
本機でのマクロ命令では、次の命令群を生成します。

		オブジェクト		命令	
IN	a, 1	7ØØØ	aaaa	PUSH	a
		7ØØØ	1111	PUSH	1
		8ØØØ	ØØØØ	CALL	Ø
		1244	ØØØ2	LEA	4, 2, 4

```
OUT  a,1    7000 aaaa  PUSH a
            7000 1111  PUSH 1
            8000 0002  CALL 2
            1224 0002  LEA  4,2,4
EXIT        6400 0004  JMP  4
```

これは００００Ｈ～０ＦＦＦＨに疑似ＯＳとランタイムパッケージが存在し、００００Ｈ，０００２Ｈ，０００４ＨがそれぞれＩＮ，ＯＵＴ，ＥＸＩＴ処理ルーチンへのジャンプテーブルと仮定しています。
参考にＣＡＳＬモードのときの仮想のメモリーマップを示します。

アドレス（１６進）



1.7　特殊命令

(1)　□命令 [×*]

機能：情報処理技術者試験の空欄穴埋め問題に対応するための、空欄入力命令です。この命令をアセンブルしますと＃ØØØØが２語生成されます。シミュレーションのときには命令語の第１語が＃ØØＸＸ（ＸＸは何でもかまいません）の命令を実行後一時停止し、　”ＳＴＯＰ　ＩＮ（番地）”と表示します。このときオブジェクトの変更やレジスタ値の確認をすることができます。その後、プログラムの続行をすることができます。

(2)　ＣＡＬＬ　＃ØØØ６

機能：画面にレジスタＧＲ０～ＧＲ３の値を表示します。[▶] キーの操作でＧＲ４、ＦＲ、ＰＣを表示します。また、プリンタが接続されているときはレジスタＧＲ０～ＧＲ３の値を印字します。
この命令を用いますとプログラムのシミュレーション、デバッグ等に便利です。

2. ＣＡＳＬモードの構成

ＣＡＳＬモードは次の４つの機能から構成されます。　40頁に機能の一覧表を示します。

エディタ　　：ソースプログラムの作成や編集をおこなうことができます。
またＣＥ－126Ｐをつなげば、ソースリストのプリントアウトができます。
またＣＥ－126ＰとＣＥ－152又はＣＥ－127Ｒ、ＣＥ－124とＣＥ－152またはＣＥ－127Ｒを接続すると、ソースプログラムのカセットへのセーブ、ロードなどが可能となります。

アセンブラ　：ソースプログラムをオブジェクトコードに変換してメモリに書き込みます。
ＣＥ－126Ｐを接続すると、アセンブルした結果をプリントアウトすることができます。

モニタ　　　：レジスタの内容や条件コードを表示させたり、書き換えることができます。
ブレークポイントのアドレスを設定することもできます。
アセンブル結果のオブジェクトコード、またはＤＳ命令等で確保したメモリの内容の表示や変更ができます。

シミュレータ：オブジェクトプログラムを実行します。
ノーマル実行とトレース実行の設定ができ、ＣＥ－126Ｐをつなげば、１ライン実行ごとに実行結果をプリントアウトできます。
モニタ機能で指定したブレークポイントでの実行中断をおこなうことができます。

2.1　プログラム作成手順

ＣＡＳＬモードでのプログラミングは、つぎの手順で作成されます。

(1)　エディタ機能を使用してソースプログラムを入力します。
また必要であればソースプログラムのプリントアウト、カセットへのセーブをおこないます。

(2)　アセンブラ機能によりオブジェクトコード（オブジェクトプログラム）を生成します。

(3)　モニタ機能を使用して、オブジェクトプログラム実行に必要なブレークポイントのアドレス設定やレジスタ、プログラムカウンタ等の設定をおこないます。

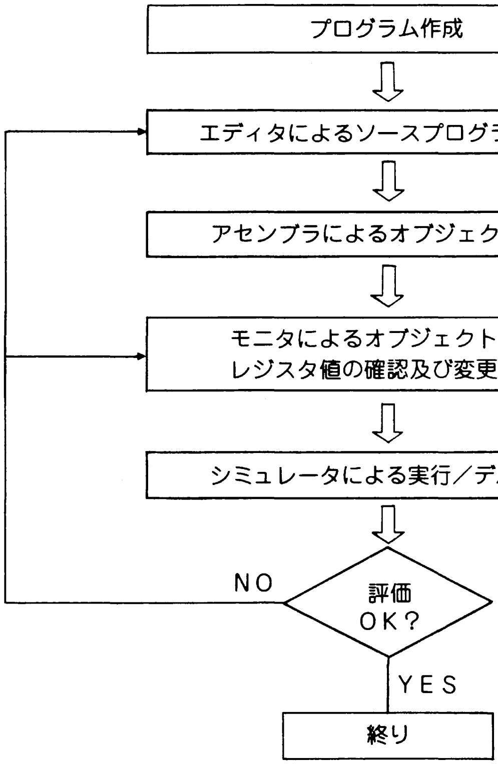
(4)　シミュレータ機能でオブジェクトプログラムのノーマル実行、またはトレース実行をさせます。

(5)　再びモニタ機能を使用して、汎用レジスタやフラグレジスタの変化を確認します。またオブジェクトコードやメモリ内容の確認と変更ができます。

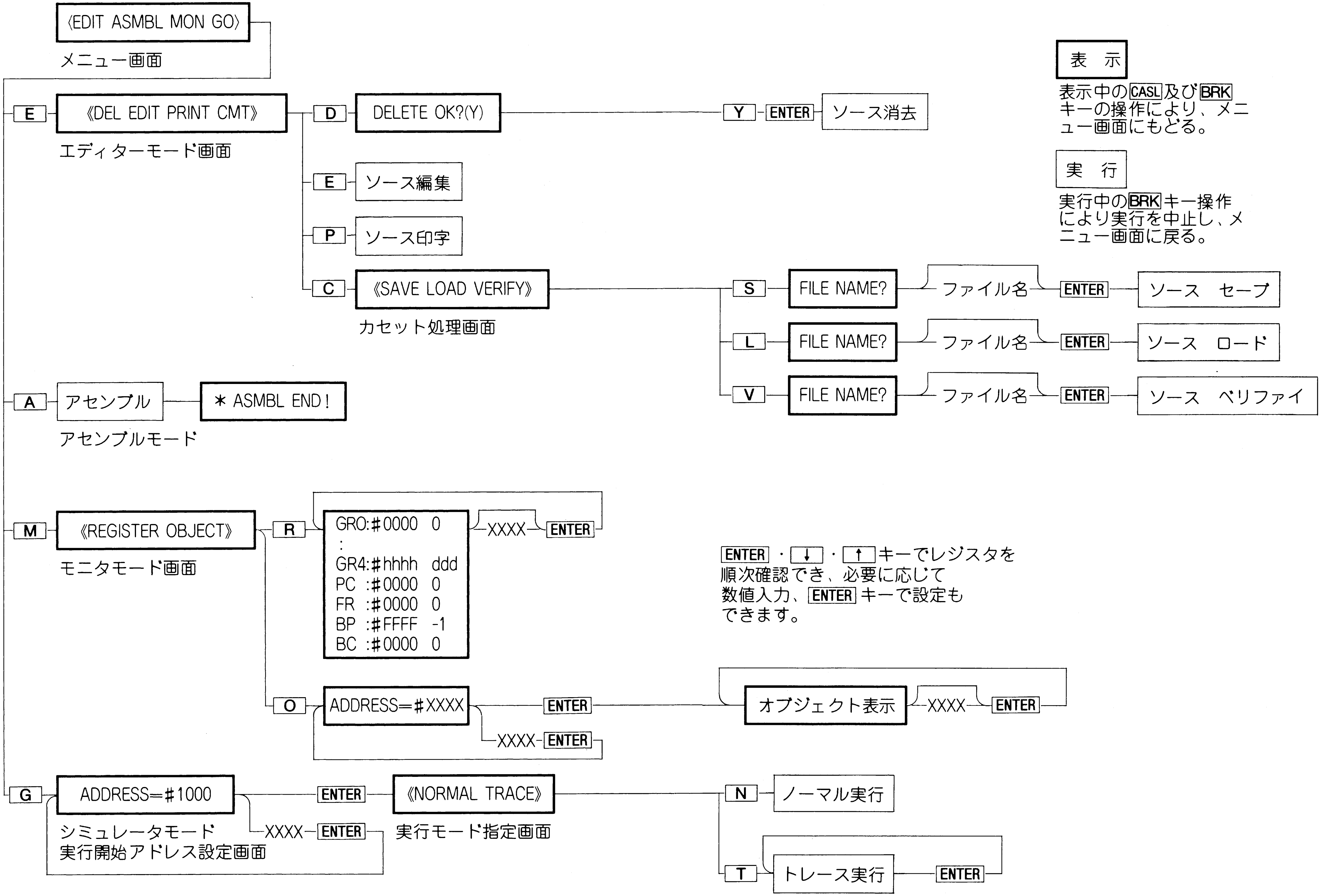
(6)　必要に応じて(3)〜(5)を繰り返します。

(7)　プログラムに誤まりが発見されたときは(1)に戻り、ソースプログラムの変更をおこない、再び(2)以降を実行します。

(8)　必要に応じて(1)〜(7)を繰り返します。



CASLモード 機能一覧表



3. CASLモードの実行

LOGICモード、PROモード、RUNモードのとき [SHIFT] [BASIC] と操作することによってCASLモードとなり、次の画面が表示されます。

[SHIFT] [BASIC] (CASL)

```
〈EDIT ASMBL MON GO〉
```

この表示画面を「メニュー画面」と呼びます。この表示状態で、以下のキー操作が可能となり、それぞれのモードに移ります。またどのモードからも [BRK] キー操作でメニュー画面表示に戻ります。

操　作	動　作
[E]	エディタモードになり、エディタ画面が表示されます。
[A]	アセンブルモードになりアセンブルが実行されます。
[M]	モニタモードになりモニタ画面が表示されます。
[G]	シミュレータモードになり、実行開始アドレス設定画面が表示されます。

4. エディタモード

メニュー画面表示で [E] キーを押すとエディタ画面が表示されます。

〈EDIT　ASMBL　MON　GO〉

[E]　　《DEL　EDIT　PRINT　CMT》

エディタモードは主にソースプログラムの作成や更新に使用されます。
エディタ画面が表示された状態のとき、以下のキー操作が可能になります。

操　作	機　　能
[D]	ソースプログラムを消去します。
[E]	ソースプログラムの入力、修正や削除等の編集をおこないます。
[P]	ソースリストのプリンタへの出力をおこないます。
[C]	ソースプログラムのカセットへのセーブ、カセットからのロードまたはベリファイをおこないます。

4.1　ソースプログラムの入力形式

ソースプログラムの構成を以下に示します。

32767	BGN	ADD	Ø,DAT,1	;SAMPLE
ラインNo.欄	ラベル欄（省略可）	命令コード欄	オペランド欄	注釈欄（省略可）

(1)　ラインNo.欄

① ラインNo.は、1～32767までの数値を使用できます。

② 1～32767の範囲外の値を指定し [ENTER] キーを押すと ＊SYNTAX　ERR＊と表示されます。この場合 [BRK] キーを押すと、メニュー画面に戻りますので、ソースプログラムの編集を継続する場合、[E] [E] とキーを押しエディタモードに戻す必要があります。[C·CE] キーを押すと、エディタモードのままプロンプト（>）を表示します。
この場合、[↓]、[↑] キーを押すと、入力した内容を再表示します。

③ [C·CE] キーを押してからの [↑] キーで最終行が、また、[×＊] 行番号 [ENTER] キーを押すと入力した番号のラインが表示できます。

(2)　ラベル欄

① ラベルは1～6文字で、7文字以上の文字を入力した場合、先頭から、6文字がラベルとして有効となります。

② CASLではラベルは6文字以内で先頭の文字はアルファベット大文字、2文字目以降はアルファベット大文字または数字と定義されています。
本機では、コンマ（,）とスペースを除いた記号ををラベルとして使用することができます。なおラベル欄にセミコロン（;）を書いて注釈ラインとすることもできます。

③ ラベルは、ラインNo．に続けて入力する必要があります。

④ ラベル入力後 SPC キーを操作し、命令コード入力をおこないます。
SPC キーは、ラベル入力後、ならびに命令コード入力後の操作では、TABの動作をおこない、それぞれの入力位置にカーソルが移動します。

(3)　命令コード欄

① クイックコマンドキー（ SHIFT キーを押した後それぞれのキーを押す）による直接入力ができます。この場合、命令コード欄とオペランド欄の間のスペースは自動的に確保されます。

② アルファベットキーで命令コードを入力する場合、入力後 SPC キーを押してTABの動作を実施させる必要があります。

(4)　オペランド欄

① オペランド欄はGRフィールド、adrフィールド、XRフィールドで構成されます。

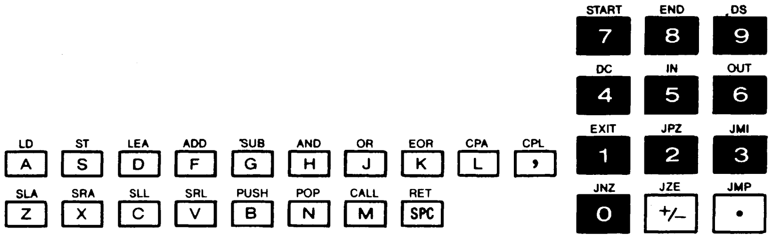
② 各ブロックの区切りには、必ずコンマ（,）が必要です。

③ XRフィールドは省略することができます。

(5)　注釈欄

① プログラムに注釈を付加する場合、オペランドの直後にセミコロン（;）を入力し、その後に注釈を入力することができ、プログラムに注釈をつけることができます。注釈欄を含めて1行に最大80文字まで入力することができます。

クイックコマンドキー配列（ SHIFT キー操作必要）



4.2　ソースプログラムの消去

エディタ画面で [D] キーを押すと下記表示となり、キーの入力待ちとなります。

[E]	《DEL　EDIT　PRINT　CMT》
[D]	DELETE　OK?(Y)_

ここで [Y] [ENTER] と入力するとソースプログラムを消去してメニュー画面に戻ります。
他のキーと [ENTER] キーまたは [ENTER] キーのみを入力した場合は何もせずにエディタ画面に戻ります。

4.3　ソースプログラムの編集

エディタ画面で [E] キーを押すと、ソースプログラムの入力、編集が可能になります。 [E] キーが押されたとき、ソースプログラムが格納されていないときにはプロンプト（〉）が表示されます。

	〈EDIT　ASMBL　MON　GO〉
[E]	《DEL　EDIT　PRINT　CMT》
[E]	〉

ソースプログラムが格納されているときには、先頭の行の内容が表示されます。

	〈EDIT　ASMBL　MON　GO〉
[E]	《DEL　EDIT　PRINT　CMT》
[E]	1Ø:L1　START　L2

編集機能を使用すると、ソースプログラムが格納されていない状態では、プログラムの新規入力ができ、格納されている状態では、プログラムの修正ができます。
また、格納されているソースプログラム全体を消去する必要のあるときは [BRK] キーでメニュー画面に戻り、再び [E] キーによりエディタモードにしてから、**4.2** のソースプログラム消去を実行してください。
次にソースプログラムの入力の手順を説明します。

(1) ラインNo.を入力します。

(2) ラベルがないときは、[SPC] キーを押すと自動的にカーソルが命令コード欄に移ります。ラベルを入力するときは、ラインＮｏ．に続けて入力して [SPC] キーを押します。
この空白はラベル欄と命令コード欄の間の区別をするために必要です。 [SPC] キーにより自動的にＴＡＢ動作（カーソルを命令コード欄まで移す）がおこなわれます。

(3) 命令コードはクイックコマンドで入力でき、入力後カーソルは自動的にオペランド欄に移ります。
命令コードはアルファベットキーでも入力できますが、入力後 [SPC] キーを入力する必要があります。 [SPC] キーにより自動的にＴＡＢ動作がおこなわれます。

(4) オペランドは、コンマ（,）で区切って入力します。

(5) 注釈は、オペランドの直後にセミコロン（;）を入力して、その後にコメントを入力します。ただし、ＤＣ命令の文字列中のセミコロンは注釈のための区切りとしては扱われません。（文字として扱われます。）このコメントは省略できます。

(6) １行を入力完了したら [ENTER] キーを押して、プログラムをメモリに格納します。 [ENTER] キーを押すとラインＮｏ．欄とラベル欄の間にコロン（：）が表示されて次のラインの入力が可能になります。

(7) 空欄穴埋め問題は＊命令を [□命令 ×＊] キーで入力します。

(8) すべてのラインの入力が完了したら、 [BRK] キーでメニュー画面に戻ります。

〈例〉ソースプログラムの入力

操作	表示
[E] を操作	〉
[1] [0] [L] [1] [SPC]	1ØL1 _
[SHIFT] [7] (START)	1ØL1 START _

キー操作	表示
[L] [2]	1ØL1　START　L2＿
[ENTER]	1Ø:L1　START　L2
[2] [0] [L] [2] [SPC]	2ØL2　＿
[SHIFT] [A] (LD)	2ØL2　LD　＿
[G] [R] [1] [,] [A]	2ØL2　LD　GR1, A＿
[ENTER]	2Ø:L2　LD　GR1, A

以下同様の操作で引き続き右のプログラムを入力してください以降の説明で本プログラムを活用します。
全ラインの入力完了後、[BRK]キーでメニュー画面に戻ります。

```
30          ADD     GR1, B
40          ST      GR1, C
50          EXIT
60   A      DC      10
70   B      DC      20
80   C      DS      1
90          END
```

（注）START命令入力時、[S] [T] [A] [R] [T] とキー操作するときは、[T] のつぎに [SPC] キーをおさなければなりません。LDなどCASLの他の命令入力時も同じです。

4.4　ソースプログラムの修正

ソースプログラムがメモリに格納されているとき、エディタ画面で [E] キーを押すとプログラムの先頭の行の内容が表示されます。
ソースプログラムを修正するためには、修正したいラインNo.のプログラムを表示させて、カーソルを修正する場所へ移動させる必要があります。このためには次のキーが利用できます。

[↓] キー

表示されているラインNo.の次のラインNo.のプログラムが表示されます。
最後部のラインNo.のプログラムが表示されているときに、このキーが押されても表示は変化しません。

[↑] キー

表示されているラインＮｏ．の１つ前のラインＮｏ．のプログラムが表示されます。
先頭のラインＮｏ．のプログラムが表示されているとき、このキーが押されても表示は変化しません。

[C·CE] キー

画面はプロンプト（>）の表示となります。この後 [↓] キーを押すと先頭のプログラムラインＮｏ．が、[↑] キーを押すと最後のラインＮｏ．が表示されます。
長いプログラムで先頭、または、最後のラインＮｏ．を表示させたい場合、[C·CE] キーを利用すると便利です。
（注）[↓]、[↑] キーは押し続けると動作が継続的に実行されます。

[▶] キー

カーソルが表示され、カーソルを１文字ずつ右へ移動します。カーソル表示中に [SHIFT] [▶] と押すと、スペースの右側の文字または最後部の文字の右側へカーソルが移動します。

[◀] キー

カーソルが表示され、カーソルを１文字ずつ左へ移動します。カーソル表示中に [SHIFT] [◀] と押すと、スペースの右側の文字または先端の文字へカーソルが移動します。

[DEL] キー

カーソルの位置にある１文字が削除され、カーソルより右側にある文字は１字左に詰められます。

[BS] キー

カーソルの左側にある１文字が削除され、カーソルより右側にある文字は１字左に詰められます。

[INS] キー
文字を追加したいときに使用され、このキーが押されるとカーソルのある位置にインサートマーク（⊏⊐）が表示されます。カーソルがあった位置の文字から右側の文字は1文字右へ移動され、カーソルはインサートマークのある位置で点滅します。次のキーでインサートマークがある位置に文字が入力できます。

[X*] キー
プロンプト（>）表示中、またはカーソルが表示されていないときに [X*] ラインNo. [ENTER] と操作すると、任意のラインNo. のプログラムを表示することができます。なお一致するラインNo. がないときは、最も近い上位のラインNo. のプログラムを表示します。

（注）文字の追加、削除の操作は [ENTER] キーが押されなければ、格納されているプログラムは変更されません。したがって文字の追加、削除の操作をおこなっても [ENTER] キーを押す前に、[↑] キーまたは [↓] キーで他のラインNo. のプログラムを表示させ、再び [↑] キーまたは [↓] キーでもとのラインNo. のプログラムを表示させると該当のラインNo. のプログラムは追加、削除の操作のおこなわれる前の内容になっています。
[ENTER] キーは修正の正しいことを確認して操作してください。

〈例〉表示データの変更
今までに説明しました方法で、[↓]、[X*] などのキーを操作して修正するラインNo. を表示してください。

操作	表示
	60：A　　DC　　10
[▶][▶]…[▶]（または [▶] キーを押しつづける）	60：A　　DC　　10（カーソルをこの位置へ）
[3]（0を3に書き換える）	60：A　　DC　　13_
[◀][INS]	60：A　　DC　　1⊏⊐3
[2]（1と3の間に2を挿入）	60：A　　DC　　123

BS （2を削除）

```
60：A      DC      13
```

DEL （3を削除）

```
60：A      DC      1_
```

（注）修正は ENTER キーの入力ではじめて格納されているプログラムが書き換えられ、変更が完了します。したがってここまでの操作では６０ラインのプログラムは変更されていません。

4.5　ソースプログラムラインの追加、削除

ソースプログラムはラインＮｏ．の小さい方から大きい方への順序で並べられます。ここではソースプログラムにラインを追加または削除する方法を説明します。

(1) プログラムを入力するとき、１ライン抜かしてしまったり、新しいラインを追加したいときは、次の方法で追加してください。

〈例〉プログラムに、次のようにＳＬＡ　ＧＲ１，１を追加したい場合

```
10：L1      START  L2
20：L2      LD     GR1，A
                            ← この間にSLA GR1，1を追加したい
30：        ADD    GR1，B
40：        ST     GR1，C
50：        EXIT
60：A       DC     10
70：B       DC     20
80：C       DS     1
90：        END
```

ラインNo.20のプログラムとラインNo.30のプログラムの間に新たなプログラムを追加する場合は、この間のラインNo.（21〜29のいずれか）を使って、プログラムを書きます。たとえばラインNo.を25として、

25 SPC ＳＬＡ SPC ＧＲ１，１ ENTER

と入力します。

プログラムは次のようになっています。（ ↓ キーおよび ↑ キーで確認してください。）

```
10:L1    START L2
20:L2    LD    GR1,A
25:      SLA   GR1,1
30:      ADD   GR1,B
40:      ST    GR1,C
50:      EXIT
60:A     DC    10
70:B     DC    20
80:C     DS    1
90:      END
```

(2)　不要になったラインNo．のプログラムを削除したいときは、削除するラインNo．を入力して ENTER キーを押します。

〈例〉上の例で追加したラインNo．25のプログラムを削除したい場合

2 5 ENTER

と入力します。

（注）

●削除がおこなわれるとプロンプト（〉）が表示されます。

●入力されたラインNo．を持つプログラムがない場合は、削除はおこなわれずにプロンプト（〉）が表示されます。

メニュー画面に戻るには BRK キーを押します。

5. ソースプログラムリストのプリントアウト

本機にプリンタが接続されているときは、ソースプログラムのリストをプリンタに印字させることができます。
プリンタは、CE－126Pが使用できます。
ソースプログラムのリストを印字させるときは、メニュー画面で [E] キーを押してエディタ画面とします。次に [P] キーを押すとソースプログラムの印字がおこなわれます。

```
<EDIT  ASMBL  MON  GO>
```

[E] [P]

```
<<DEL  EDIT  PRINT  CMT>>
```

プリンタが接続され、プリンタの電源が入っているとき、ソースプログラムが印字されます。印字が終了するとメニュー画面に戻ります。
プリンタが接続されていないとき、またはプリンタの電源が入っていないときには、何も実行されません。
〈例〉ソースプログラムリストは次の形式で印字されます。

```
10:L1   START L2
20:L2   LD    GR1,A
30:     ADD   GR1,B
40:     ST    GR1,C
50:     EXIT
60:A    DC    10
70:B    DC    20
80:C    DS    1
90:     END
 ①   ②     ③      ④
```

① 行番号
② ラベル（3文字以上の場合、以降が後方へずれます。）
③ 命令コード
④ オペランドおよび注釈
（全体で24桁を超えた部分は次の行に印字されます。）

プリンタを使用するときは次の事項に注意してください。
(1) ソースプログラムリストの1行が24桁を超えるときは、25桁目以降は次の行に印字されます。
(2) プリンタが印字中は、BUSYシンボルが表示され、印字が終了すると、メニュー画面に戻ります。
(3) プリンタが印字中にプリンタの紙づまりや、電源のローバッテリ状態が発生すると、しばらくした後、メニュー画面にもどります。

6. ソースプログラムのテープレコーダへのセーブ・ロード

ここではテープレコーダを使用して、ソースプログラムをテープにセーブしたり、テープからロードしたり、あるいはセーブした内容を照合（ベリファイ）したりする方法を説明します。
テープレコーダを接続して、エディタ画面で C キーを押すとカセット処理画面が表示されます。

C

```
<<SAVE  LOAD  VERIFY>>
```

カセット処理画面では、次のキー操作が可能となります。

操　作	機　能
S	ソースプログラムをテープに記録します。
L	テープに記録されたソースプログラムをロードします。
V	テープに記録されたソースプログラムと計算機に記憶されたソースプログラムの照合をおこないます。

カセット処理中の停止は BRK キーでおこないます。

6.1 ソースプログラムのセーブ

ソースプログラムをテープに記録するときは、カセット処理画面で S キーを押します。ファイル名を聞いてきますので、最大７文字までのファイル名を入力して ENTER キーを押すとソースプログラムはテープにセーブされます。

```
<<SAVE  LOAD  VERIFY>>
```

S

```
FILE  NAME? _
```

またファイル名を入力せずに ENTER キーを押すとファイル名のないプログラムがテープにセーブされます。

6.2 ソースプログラムのロード

テープに記録されているソースプログラムをロードするときは、カセット処理画面で [L] キーを押します。 [L] キーを押すと、ファイル名を聞いてきます。ファイル名を入力して [ENTER] キーを押すと、ファイル名の一致するプログラムを見つけてロードします。プログラムのロードがおこなわれると画面の右端に＊記号が表示され、ロードが終了すると＊記号が消えます。
またファイル名を入力せずに [ENTER] キーのみを押すと、テープの現在位置から最初に見つかったプログラムがロードされます。

```
<<SAVE  LOAD  VERIFY>>
```

[L]

```
FILE  NAME? _
```

6.3 ソースプログラムの照合

テープにセーブされたソースプログラムと計算機に記憶されているプログラムとの照合をおこなうためには、カセット処理画面で [V] キーを押します。 [V] キーが押されるとファイル名を聞いてきます。ファイル名を入力して [ENTER] キーを押すとファイル名の一致するプログラムを見つけて照合します。また、ファイル名を入力せずに [ENTER] キーのみが押されると、テープの現在位置から最初に見つかったプログラムが照合されます。

```
<<SAVE  LOAD  VERIFY>>
```

[V]

```
FILE  NAME? _
```

セーブされたプログラムが見つかると、照合中は画面の右端に＊記号が表示され、照合の実行が完了するとメニュー画面に戻ります。

6.4 エラー表示

次のような場合、下記のエラー表示をおこないます。

●プログラムロード中、誤ったデータを入力した場合

●ベリファイ結果が一致しなかった場合

```
*CMT  ERROR*
```

●ロード途中でプログラムを記憶するためのメモリ容量が不足した場合

```
*MEM  ERROR*
```

（注）エラーを解除する場合、C·CE キーを押すとエディタ画面に、BRK キーを押すとメニュー画面に戻ります。

7. アセンブル

ソースプログラムをアセンブルしてオブジェクトコードに変換するためにはメニュー画面で [A] キーを押します。
[A] キーが押されるとアセンブルが開始されます。

```
〈EDIT  ASMBL  MON  GO〉
```

[A]　　アセンブルの実行が開始されます。
アセンブルが完了すると次の表示をおこないます。

```
*  ASMBL  END  !
```

[C·CE] または [BRK] キーでメニュー画面に戻ります。

7.1 アセンブルリストをプリンタへ印字させる方法

メニュー画面で [SHIFT] [ENTER] キーを押すと、プリンタが接続されているときＰＲＩＮＴシンボルが点灯します。再度 [SHIFT] [ENTER] キーを押すと、ＰＲＩＮＴシンボルは消灯します。
ＰＲＩＮＴシンボルが点灯した状態で、アセンブルを実行するとプリンタにアセンブルリストが印字されます。

```
ADD :OBJECT    :LINE NO.
               :    10
1000:1010 1008:     20
1002:2010 1009:     30
1004:1110 100A:     40
1006:6400 0004:     50
1008:000A      :    60
1009:0014      :    70
100A:0000      :    80
               :    90
|-①-|-----②-----|---③---|
LABEL :ADDRESS  ┬
L1       1000   │
L2       1000   │
A        1008   ④
B        1009   │
C        100A   ┴
```

① アドレス１６進
② オブジェクト１６進
③ 対応するソースプログラムのラインＮｏ.
④ ラベルテーブル

アセンブルリストにはソースプログラムは印字されません。またアセンブルエラーが発生したときはエラーが発生したラインの前までが印字されます。

7.2 エラーメッセージ

アセンブルの実行で、ソースプログラムに以下のエラーが検出されるとエラーメッセージが表示されます。
エラーメッセージは、 C·CE または BRK キーでエラーを解除し、メニュー画面に戻ることができます。
エラー解除後、エディタモードでソースプログラムを表示すると、最初にエラーの発生したラインNo．のプログラムが表示されます。

エラーの種類	エラーメッセージ	理　由
オペコードエラー	＊OP　ERR（ラインNo.）　＊	ラインNo.のプログラムの命令コードに誤りがあります。
	＊OP　ERR（Ø）＊	ソースプログラムがありません。
オペランドエラー	＊OPR　ERR（ラインNo.）　＊	ラインNo.のプログラムのオペランドに誤りがあります。
ラベルエラー	＊LB　ERR（ラインNo.）　＊	ラインNo.のプログラムのラベルに誤りがあります。
メモリエラー	＊MEM　ERR（ラインNo.）　＊	オブジェクトコードを記憶するメモリが不足しています。
	＊MEM　ERR（Ø）＊	スタック用メモリが不足しています。（31ワード以下）
その他のエラー	＊OTHER　ERROR＊	プログラムの先頭にSTARTが、または終りにEND命令がありません。

8. モニタモード

モニタモードでは、仮想計算機ＣＯＭＥＴの各レジスタの内容の表示および値の設定をおこなう機能と、メモリに格納されたオブジェクトコードの表示や設定をおこなう機能があります。
シミュレータモードのためのブレークポインタの内容表示と設定も、モニタモードでおこないます。
モニタモードにするためには、メニュー画面状態で [M] キーを押すことによりおこなわれ、モニタ画面が表示されます。

```
<EDIT  ASMBL  MON  GO>
```

[M]

```
<<REGISTER  OBJECT>>
```

モニタ画面表示状態では、次の操作が可能です。

操　作	機　　能
[R]	レジスタの内容の表示と、レジスタへの値の設定ができます。
[O]	オブジェクトコードを逆アセンブルした命令コードと共に表示します。

8.1 レジスタの内容の表示

モニタ画面で [R] キーを押すとレジスタの内容表示とブレークポインタの内容がレジスタ名、内容の１６進と１０進数表示の順に表示されます。

```
<<REGISTER  OBJECT>>
```

[R]

```
GR0：#0000  0
```

レジスタ　16進数で内容の表示　10進数で内容の表示

[ENTER]

```
GR1：#0000  0
```

以後 [ENTER] キーを押すごとに次のレジスタの内容が表示されます。また、[↑]、[↓] キーで前後のレジスタの表示ができます。メニュー画面に戻るには[BRK]キーを押します。

[ENTER]	GR2：#ØØØØ　Ø
[ENTER]	GR3：#ØØØØ　Ø
[ENTER]	GR4：#1E1F　7711
[ENTER]	PC　：#1ØØØ　4096
[ENTER]	FR　：#ØØØØ　Ø
[ENTER]	BP　：#FFFF　65535
[ENTER]	BC　：#ØØØØ　Ø

さらに [ENTER] キーを押すと、GR0レジスタの表示にもどります。
各レジスタ名は以下のとおりです。

表　示	レジスタ名	機　　　能
GR0〜GR4	汎用レジスタ	算術演算と論理演算に用います。GR1〜GR4は指標レジスタとして、また、GR4はスタックポインタとしても用います。
PC	プログラムカウンタ	次に実行すべき命令が格納されているアドレスを記憶します。
FR	フラグレジスタ	特定の命令を実行した結果が、正、ゼロ、負になると、それぞれ0、1、2にセットされます。
BP	ブレークポインタ	シミュレータモードでの実行を制御するために使用します。
BC	ブレークカウンタ	

8.2 レジスタに値を設定する方法

GR0～GR4、PC、FR、BP、BCの各レジスタに値を設定することができます。値を設定しようとするレジスタを表示している状態で、値を10進、16進ラベル、文字などで入力して、ENTER キーを押すとレジスタに値が設定されます。

10進数の入力：	123	ENTER	#ØØ7B 123が入力
16進数の入力：	#ØØ7B	ENTER	#ØØ7B 123が入力
ラベルの入力 ：	”L1”	ENTER	ラベル”L1”の定義されているアドレスが入力
文字の入力 ：	'A'	ENTER	Aの文字コード#ØØ41 65が入力

設定終了後 BRK キーでメニュー画面に戻ります。

レジスタに値を設定するときは次の点に注意してください。

(1) 設定する値は10進数、16進数、ラベル、文字で入力します。

(2) 負数は先頭に － キーでマイナスを入力します。

(3) 設定できる値の範囲は−32768～65535までの整数が可能です。この範囲を超えた値を入力して ENTER キーを押すと入力前の値に戻ります。

(4) FRの値は0，1，2しかとれません。0，1，2以外の値が入力されると他のビットは無視されます。表示は入力された値を表示しています。

(5) アセンブル実行後の各レジスタには、以下の値が設定されています。

GRØ～GR3	：Ø
GR4	：オブジェクトエリアの最上位番地＋1
PC	：メインプログラムの実行開始アドレス（STARTで指定されたラベルのアドレス）
FR	：Ø
BP	：FFFF（16進）、65535（10進）（ブレークポインタが設定されていない状態。）
BC	：Ø

8.3　オブジェクトコードの表示

アセンブルの実行によって、作成されたオブジェクトプログラムの内容を表示させることができます。このためにはモニタ画面で [O] キーを押します。

[O] キーが押されると、オブジェクトプログラムが記憶されているときは、アドレス#1000が表示されます。

```
<<REGISTER OBJECT>>
```

[O]

```
ADDRESS=#1000
```

メインプログラムの開始アドレス

次に [ENTER] キーを押すと、オブジェクトプログラムエリアの#1000番地に記憶されたオブジェクトコードと逆アセンブルしたニーモニックが表示されます。

[ENTER]

```
1000:1010 LD    1,#1008
```

アドレス　オブジェクトコード　オペランド　GRフィールド　アドレスフィールド　XRフィールド

ニーモニック表示

以後、[ENTER] キーを押すたびに、アドレスが1番地ずつ増加し、そのアドレスに記憶されている内容が表示されます。
命令語の第2語目を表示するときは逆アセンブルされません。

[ENTER]

```
1001:1008
```

オブジェクトプログラムが記憶されていないときに [O] キーを押すと次のエラー表示となります。

[O]　　　＊ OBJECT ERR ＊

このときは、[CCE]　または [BRK] キーでメニュー画面に戻します。
（注）次のような場合、メモリに記憶されたオブジェクトプログラムは消去されます。
●エディタモードで [E] キーを押してソースプログラムを編集した場合
●ＬＯＧＩＣまたはＢＡＳＩＣモードへモードが切り替わった場合
このような場合には、メニュー画面で [A] キーを押して再度アセンブルをおこなうと、オブジェクトプログラムが作成されます。

8.4　特定のアドレスに記憶された内容の表示

メインプログラムの開始アドレスを表示している画面で、表示させたいアドレスを１０進または１６進、ラベルで入力し [ENTER] キーを押しますと、アドレスを１６進で表示します。続く [ENTER] キーで内容が表示されます。入力されたアドレスが偶数であれば、偶数番地を命令語の第１語目、次の番地を第２語目とみなして逆アセンブルします。奇数が入力されたときは奇数番地を第１語目、偶数番地を第２語目とみなします。

	<<REGISTER OBJECT>>
[O]	ADDRESS=#1000
#1006 [ENTER]	ADDRESS=#1006
[ENTER]	1006:6400 JMP #0004

メニュー画面に戻るには [BRK] キーを押します。
（注１）オブジェクトコードが記憶されていないアドレスを指定した場合オブジェクトコードは”０００　＊”が表示されます。

1010:0000 *

（注２）入力できるアドレスは＃１０００からアセンブル終了時ＧＲ４で示されるアドレス－１の範囲です。

9. シミュレータモード

オブジェクトコードを実行させるモードです。
メニュー画面で、 G キーを押すとオブジェクトプログラムの実行処理に移ります。

```
<EDIT  ASMBL  MON  GO>
```

G

```
ADDRESS=#1ØØØ
```

アセンブルを実行した後に、このモードに移ったとき、ＳＴＡＲＴ命令で指定された実行開始アドレスが表示されます。実行アドレスが指定されていないときは＃１ØØØが自動的に設定されて表示されます。
実行開始アドレスの設定を変更するときは、この状態でアドレスを１０進、１６進数、またはラベルで入力します。
次に ENTER キーを押すと、実行モード指定画面になります。

ENTER

```
<<NORMAL  TRACE>>
```

実行モード指定には次の操作ができます。

操　作	機　　能
N	ノーマル実行
T	トレース実行

9.1 ノーマル実行

実行モード指定画面で N キーを押すと、設定されたスタートアドレスからプログラムの実行を開始します。

```
<<NORMAL  TRACE>>
```

N　　プログラムが実行されＥＸＩＴ命令の実行後、メニュー画面に戻ります。

```
<EDIT  ASMBL  MON  GO>
```

ノーマル実行中、つぎの場合にプログラムの実行を停止して、停止したアドレスを表示します。

●ブレークポインタ（BP）が設定されているとき、プログラムカウンタ（PC）の値がブレークポインタと一致した場合。この場合、ブレークポインタが設定されているアドレスの命令は実行されていませんので注意してください。
● [BRK] キーが押された場合。この場合、プログラムが中断（ブレーク）したアドレスが入っていますので、 [ENTER] キーを押すかまたは、[C·CE] 、[BRK] キーでメニュー画面に戻り、再び [G] キーを押すと、プログラムの継続ができます。
● ＊ 命令を実行した場合
（命令語 第1語目：＃ØØXX、第2語目：＃XXXX）を実行した場合プログラムカウンタには、命令語の第1語目のアドレスが入っていますのでオブジェクトを修正せずに、または実行アドレスを修正せずにプログラムを実行しますと、再度同じアドレスでプログラムが停止します。このような場合、命令語または実行開始アドレスを修正した後、プログラムを実行してください。
＊ 命令により停止し、停止アドレスが表示されているときに [ENTER] キーを押すと、 ＊ 命令をスキップし次の命令を実行します。

メニュー画面で [SHIFT] [ENTER] キーを押しPRINTシンボルが点灯しているときは、ノーマル実行で1命令を実行するごとに以下のフォーマットで実行結果が16進数で印字されます。

```
ADD :GR0  GR1  GR2  GR3
1000:0000 000A 0000 0000
1002:0000 001E 0000 0000
1004:0000 001E 0000 0000
1006:0000 001E 0000 0000
|-①-||--②--||--③--||--④--||--⑤--|
```

① アドレス
② GRØ
③ GR1
④ GR2
⑤ GR3

9.2 トレース実行

実行モード指定画面で、 [T] キーが押されるとスタートアドレスからプログラムのトレース実行が開始されます。
トレース実行中は1命令実行ごとに、実行結果を表示して停止します。

	‹‹NORMAL TRACE››
[T]	1ØØØ:ØØØØ ØØØA ØØØØ ØØØØ

次に [ENTER] キーを押すと次の命令を実行し、実行結果を表示します。

[ENTER]

```
1002:0000 001E 0000 0000
```

実行結果が画面の桁数を超えるとき、[◀] 、[▶] キーでスクロールして見ることができます。表示はすべて16進表示で左からアドレス：GRØ GR1 GR2 GR3 GR4 FR PCの順に表示します。
また、PRINTシンボルが点灯しているときは、実行結果をプリンタに印字することができます。

9.3 シミュレータモードでのエラー表示

シミュレータモードでは以下のエラーが検出されるとエラー表示がおこなわれます。
ここで、[C·CE] または [BRK] キーを押すとメニュー画面に戻ります。

エラーメッセージ	理　由
＊OBJECT ERR＊	オブジェクトプログラムがありません。
＊MEM ERR（アドレス）＊	●JMP命令等で使用可能な領域を超えたアドレスへのジャンプがおこなわれたとき。 ●使用可能な領域を超えて、データのロード、セーブをしようとしたとき。
＊OPR ERR(#ØØØ2)＊	OUT命令で出力文字を81文字以上にしたとき。

10. ＣＡＳＬモードの実行例

次のプログラムはラインＮｏ．130からラインＮｏ．170にある5個のデータの総和をラインＮｏ．120で確保した領域にラインＮｏ．90で格納するプログラムです。

```
 10:EXAM  START
 20:BGN   LEA    GR0, 0           （GR0をクリアする）
 30:      LEA    GR1, 0           （GR1をクリアする）
 40:      JMP    AGN1             （無条件にAGN1にジャンプする）
 50:AGN   ADD    GR0, DAT, GR1    （指標レジスタ（GR1）を利用し
                                   てデータを順番にGR0へ加算する）
 60:      LEA    GR1, 1, GR1      （GR1の内容に1を加えて、結果
                                   をGR1にいれる）
 70:AGN1  CPA    GR1, N           （GR1の内容とN番地の内容とを
                                   大小比較する）
 80:      JMI    AGN              （FRの値が‘負’を示していれば
                                   AGNへジャンプする）
 90:      ST     GR0, TTL         （GR0の内容をTTL番地へ格納する）
100:      EXIT                    （プログラムの実行終了）
110:N     DC     5
120:TTL   DS     1                （結果を格納するためのエリアを確
130:DAT   DC     #000C             保する）
140:      DC     #07F3
150:      DC     #0231             サンプルデータ
160:      DC     #0009
170:      DC     #000F
180:      END                     （プログラム終了）
```

このプログラムをエディタモードで入力し、アセンブルモードでアセンブルします。

10.1　オブジェクトコードの内容確認

以下の手順にしたがってモニタモードにして、オブジェクトコードの内容を確認します。

操　作	表　　示
	〈EDIT ASMBL MON GO〉
M	《REGISTER OBJECT》
O	ADDRESS=#1ØØØ
ENTER	1ØØØ:12ØØ LEA Ø, #ØØØØ
ENTER	1ØØ1:ØØØØ
ENTER	1ØØ2:121Ø LEA 1, #ØØØØ
ENTER	1ØØ3:ØØØØ
ENTER	1ØØ4:64ØØ JMP #1ØØA
ENTER	1ØØ5:1ØØA
ENTER	1ØØ6:2ØØ1 ADD Ø, #1Ø14, 1
ENTER	1ØØ7:1Ø14
ENTER	1ØØ8:1211 LEA 1, #ØØØ1, 1
ENTER	1ØØ9:ØØØ1
ENTER	1ØØA:4Ø1Ø CPA 1, #1Ø12
ENTER	1ØØB:1Ø12
ENTER	1ØØC:61ØØ JMI #1ØØ6
ENTER	1ØØD:1ØØ6
ENTER	1ØØE:11ØØ ST Ø, #1Ø13
ENTER	1ØØF:1Ø13
ENTER	1Ø1Ø:64ØØ JMP #ØØØ4
ENTER	1Ø11:ØØØ4
ENTER	1Ø12:ØØØ5 *
ENTER	1Ø13:ØØØØ

操 作	表 示
ENTER	1Ø14:ØØØC *
ENTER	1Ø15:Ø7F3
ENTER	1Ø16:Ø231 *****
ENTER	1Ø17:ØØØ9
ENTER	1Ø18:ØØØF *
BRK	〈EDIT ASMBL MON GO〉

10.2 ノーマル実行

次にソースプログラムのラインNo．8Ø： JMI AGNにブレークポイントを設定してプログラムの実行結果を調べることにします。
前項10.1の方法でオブジェクトコードを調べると、この命令はアドレス＃１００C番地に格納されていることがわかります。

操 作	表 示	説 明
	〈EDIT ASMBL MON GO〉	
M	《PRGISTER OBJECT》	モニタモードにします。
R	GRØ:#ØØØØ Ø	
ENTER ⋮	GR1:#ØØØØ Ø	ブレークポインタを表示させます。
ENTER	BP :#FFFF 65535	ブレークポインタは設定されていません。
# 1 0 0 C	BP :#FFFF #1ØØC_	
ENTER	BP :#1ØØC 41Ø8	ブレークポインタをアドレス#1ØØCに設定します。
ENTER	BC :#ØØØØ Ø	

操　作	表　　示	説　　明
2	BC　:#ØØØØ　2_	ブレークカウンタに2を設定します。
ENTER	BC　:#ØØØ2　2	
BRK	〈EDIT　ASMBL　MON　GO〉	メニュー画面に戻します。
G	ADDRESS=#1ØØØ	
ENTER	《NORMAL　TRACE》	
N	* STOP　IN　(#1ØØC) *	ノーマル実行し、#1ØØC番地でブレークがかかります。注1）注2）参照
BRK	〈EDIT　ASMBL　MON　GO〉	メニュー画面に戻します。
M	《REGISTER　OBJECT》	モニタモードにします。
R	GRØ:#ØØØC　12	ADD　Ø, DAT, 1で汎用レジスタGRØに最初のデータが加算されています。
↑	BC　:#ØØØØ　Ø	ブレークカウンタを表示します。
6 ENTER	BC　:#ØØØ6　6	6を設定します。注3）参照
BRK	〈EDIT　ASMBL　MON　GO〉	メニュー画面に戻します。
G	ADDRESS=#1ØØC	先程のブレークがかかった番地を表示します。
ENTER	《NORMAL　TRACE》	#1ØØC番地より実行を継続します。
N	〈EDIT　ASMBL　MON　GO〉	実行後、メニュー画面に戻ります。
M	《REGISTER　OBJECT》	
R	GRØ:#ØA48　2632	GRØに5つのデータの総和が入っています。
↓	GR1:#ØØØ5　5	GR1の値は5になっています。
BRK	〈EDIT　ASMBL　MON　GO〉	メニュー画面に戻します。

注1）プログラムカウンタ（PC）とブレークポインタ（BP）の値が一致したときに、ブレークカウンタ（BC）の値が0または1であればプログラムの実行を停止し、ブレークカウンタの値を0にします。
ブレークカウンタの値が0または1以外のときは、ブレークカウンタから1を減じて、プログラムを続行します。

注2）レジスタなどの値を確認、変更する必要がない場合は、ENTER キーを押せばプログラムを続行します。

注3）スタートアドレスとブレークポインタ（BP）の値が等しいとき、ブレークカウンタ（BC）の値が0であればプログラムを続行し、次にプログラムカウンタとブレークカウンタの値が一致したときに停止します。
また、ブレークカウンタの値が1であればなにも実行せずに停止し、ブレークカウンタの値を0にします。
ブレークカウンタの値が0および1以外のときは、ただちに1を減じて、プログラムを続行します。

10.3　ブレークポインタの解除

設定したブレークポインタの解除は、次の2通りの方法でおこなえます。

(1)　アセンブルの再実行

(2)　ブレークポインタの設定時と同じように、メニュー画面より
　　M　R　ENTER　…　ENTER　と操作し、BPレジスタを表示させせます。
　　次に－1　ENTER　とキーを押すことにより、解除できます。
　　BRK　キーでメニュー画面に戻します。

10.4　トレース実行

同じプログラムをトレースモードで実行してみましょう。
ブレークポインタは解除してください。なお、次の例はアセンブル実行直後の例を示しています。

操　作	表　　示	説　　明
	〈EDIT　ASMBL　MON　GO〉	
G	ADDRESS=#1ØØØ	
ENTER	《NORMAL　TRACE》	トレースモードにします。
T	1000：0000　0000　0000　0000	LEA　Ø，ØでGRØをリセットします。
ENTER	1002：0000　0000　0000　0000	LEA　1，ØでGR1をリセットします。
ENTER	1004：0000　0000　0000　0000	JMP　AGN1に無条件にジャンプします。▶キーを押し続け、PCの値をみれば100A番地であることが確認できます。
ENTER	100A：0000　0000　0000　0000	CPA　1，NでGR1の内容とN番地の値を比較します。
ENTER	100C：0000　0000　0000　0000	JMI　AGNでFRが2のときAGNアドレスにジャンプします。
ENTER	1006：000C　0000　0000　0000	ADD　Ø，DAT，1で指標レジスタ（GR1）を利用してデータをGRØに加算します。
ENTER	1008：000C　0001　0000　0000	GR1に1を加算します。

操　作	表　　示	説　　明
ENTER	100A：000C 0001 0000 0000	以降、データ数分繰り返します。
ENTER	100C：000C 0001 0000 0000	
ENTER	1006：07FF 0001 0000 0000	
ENTER	1008：07FF 0002 0000 0000	
ENTER	100A：07FF 0002 0000 0000	
ENTER	100C：07FF 0002 0000 0000	
ENTER	1006：0A30 0002 0000 0000	
ENTER	1008：0A30 0003 0000 0000	
ENTER	100A：0A30 0003 0000 0000	
ENTER	100C：0A30 0003 0000 0000	
ENTER	1006：0A39 0003 0000 0000	
ENTER	1008：0A39 0004 0000 0000	
ENTER	100A：0A39 0004 0000 0000	
ENTER	100C：0A39 0004 0000 0000	
ENTER	1006：0A48 0004 0000 0000	
ENTER	1008：0A48 0005 0000 0000	
ENTER	100A：0A48 0005 0000 0000	
ENTER	100C：0A48 0005 0000 0000	CPA 1, NでFRが1となりジャンプせず、次の命令を実行します。
ENTER	100E：0A48 0005 0000 0000	ST Ø, TTLでGRØの値を番地TTLへ格納します。
ENTER	1010：0A48 0005 0000 0000	EXIT命令により実行を停止します。
ENTER	〈EDIT ASMBL MON GO〉	メニュー画面に戻ります。

10.5　空欄穴埋め問題の実行例

前の例題でプログラムラインＮｏ．６０が空欄穴埋め問題となっていた場合、下の例のように６０ラインに”＊”を入力します。

```
10:EXAM START
20:BGN LEA   GR0,0
30:    LEA   GR1,0
40:    JMP   AGN1
50:AGN ADD   GR0,DAT,GR1
60:    LEA   GR1,1,GR1
70:AGN1 CPA   GR1,N
80:    JMI   AGN
90:    ST    GR0,TTL
100:    EXIT
110:N   DC    5
120:TTL DS    1
130:DAT DC    #000C
140:    DC    #07F3
150:    DC    #0231
160:    DC    #0009
170:    DC    #000F
180:    END
```

```
10:EXAM START
20:BGN LEA   GR0,0
30:    LEA   GR1,0
40:    JMP   AGN1
50:AGN ADD   GR0,DAT,GR1
60:    *
70:AGN1 CPA   GR1,N
80:    JMI   AGN
90:    ST    GR0,TTL
100:    EXIT
110:N   DC    5
120:TTL DS    1
130:DAT DC    #000C
140:    DC    #07F3
150:    DC    #0231
160:    DC    #0009
170:    DC    #000F
180:    END
```

このプログラムをアセンブルしますとラインＮｏ．６０に対応するオブジェクトコードが”＃００００　＃００００”となります。

```
ADD :OBJECT    :LINE NO.
               :   10
1000:1200 0000:   20
1002:1210 0000:   30
1004:6400 100A:   40
1006:2001 1014:   50
1008:0000 0000:   60
100A:4010 1012:   70
100C:6100 1006:   80
100E:1100 1013:   90
1010:6400 0004:  100
1012:0005      :  110
1013:0000      :  120
1014:000C      :  130
1015:07F3      :  140
1016:0231      :  150
1017:0009      :  160
1018:000F      :  170
               :  180
```

このプログラムを実行しますと、＃１００８番地でプログラムの実行を停止し、レジスタ、オブジェクトの確認、修正をすることができます。

操 作	表 示	説 明
	〈EDIT ASMBL MON GO〉	
G	ADDRESS＝#1000	
ENTER	《NORMAL TRACE》	
N	＊ STOP IN（#1008）＊	＊命令があるとプログラムの実行を停止します。
ENTER	＊ STOP IN（#1008）＊	注１）
BRK	〈EDIT ASMBL MON GO〉	
M R	GR0：#0018 24	レジスタの内容を確認することができます。
ENTER	GR1：#0000 0	注２）
BRK	〈EDIT ASMBL MON GO〉	
M O	ADDRESS＝#1000	
#1008 ENTER	ADDRESS＝#1008	
ENTER	1008:0000 ＊	
LEA GR1,1,GR1	1008:0000 LEA GR1, 1, GR1_	空欄の答えを入力します。
ENTER	1008:1211 LEA 1, #0001, 1	アセンブルし、オブジェクトを生成します。
ENTER	1009:0001	
BRK G	ADDRESS＝#1008	
"BGN"	ADDRESS＝"BGN"_	
ENTER	ADDRESS＝#1000	もう一度初めからプログラムを実行します。
ENTER N	〈EDIT ASMBL MON GO〉	
M O "TTL" ENTER	ADDRESS＝#1013	
ENTER	1013:0A48 ＊＊＊＊＊	注３）

注1）シミュレータモードで＊命令を実行し、プログラムの実行が停止しているとき、 ENTER キーを押しますと次の命令から実行を開始します。実行後、＊命令があると再び停止します。

注2）この時点でGRØには＃Ø7FF、GR1には＃ØØØ1が入っているべきですが値が異なっていることがわかります。

注3）合計を確認すると正しい値が入っています。つまり空欄に入れた答えが正しいことがわかります。この時点でオブジェクトコードは変更されていますがソースプログラムは変更されていません。

11. メモリ領域

アセンブルによってオブジェクトコードは下図の領域に格納されます。

CASLモード ソースプログラム	CASLモード オブジェクトコード	BASIC プログラム	BASIC 変　数

11.1 格納できるステップ数の計算法

ソースプログラムの入力後、BASICモードにしてMEM ENTER を実行するとオブジェクトコードの格納可能な領域の大きさがバイト単位の数で表示されます。オブジェクトコードは1ステップ16ビット（2バイト）であるため、この領域に格納できるステップ数は次式で計算できます。

$$\mathrm{INT}\left(\frac{\mathrm{MEM}-\text{ラベル数}\times 8-4}{2}\right)$$

ただし、INTはカッコ内の計算結果の整数部を値とします。

この計算式で求めた値にはスタックエリアも含まれています。
スタックエリアが31ステップ以下になるとエラーになります。

〈例〉

```
L1    START  L2
L2    LD     GR1, A
      ADD    GR1, B
      ST     GR1, C
      EXIT
A     DC     10
B     DC     20
C     DS     1
      END
```

このプログラムの場合、オブジェクトサイズは、7ステップとなります。

12. 仮想計算機ＣＯＭＥＴとの相違点

本機は通産省の設定している仮想計算機ＣＯＭＥＴと、そのアセンブラ言語の仕様に以下の相違点があります。

(1) ＳＴＡＲＴ命令

ラベルが省略できます。また、オペランドが省略された場合、＃１ØØØ番地からプログラムを実行します。

(2) ＤＣ命令

オペランドが１０進定数で、−32768〜65535の範囲にないときは、アセンブルするとエラーになります。

(3) ＩＮ命令（ＣＡＬＬ　＃ØØØØ）

ＩＮ命令を実行すると、画面左側に”？”が表示され、キーからの入力が可能となります。このとき [↓] キー入力があると入力文字長に−１（＃ＦＦＦＦ）をセットします。

(4) ＯＵＴ命令（ＣＡＬＬ　＃ØØØ2）

ＯＵＴ命令を実行するとプリンタが接続されているときは印字し、接続されていないときは画面に表示します。

出力文字長が８１文字以上のときはエラーとなります。出力データの第８〜１５ビットの値が＃２Ø〜＃５Ｆ以外のときは、＃２Ø（スペース）として出力します。

(5) 複数プログラムの連結

個々にＳＴＡＲＴ命令とＥＮＤ命令で囲まれた複数のプログラムを入力し、アセンブルするとき、他のプログラムのラベルと同じものがあるとエラーになります。

ＳＴＡＲＴ命令により実行開始アドレスが複数定義されているとき、最後の指定が有効となります。ただし、実行開始アドレスはプログラム実行時に変更することができます。